



LECTURE 29

PROCESS MANAGEMENT IN MACH

History of Mach

- Mach's earliest roots go back to a system called RIG (Rochester Intelligent Gateway), which began at the University of Rochester in 1975. Its main research goal was to demonstrate that operating systems could be structured in a modular way.
- When one of its designers, Richard Rashid, left the University of Rochester and moved to Carnegie-Mellon University in 1979, he wanted to continue developing message-passing operating systems but on more modern hardware. The machine selected was the PERQ. The new operating system for the PERQ was called Accent. It is an improvement of RIG.

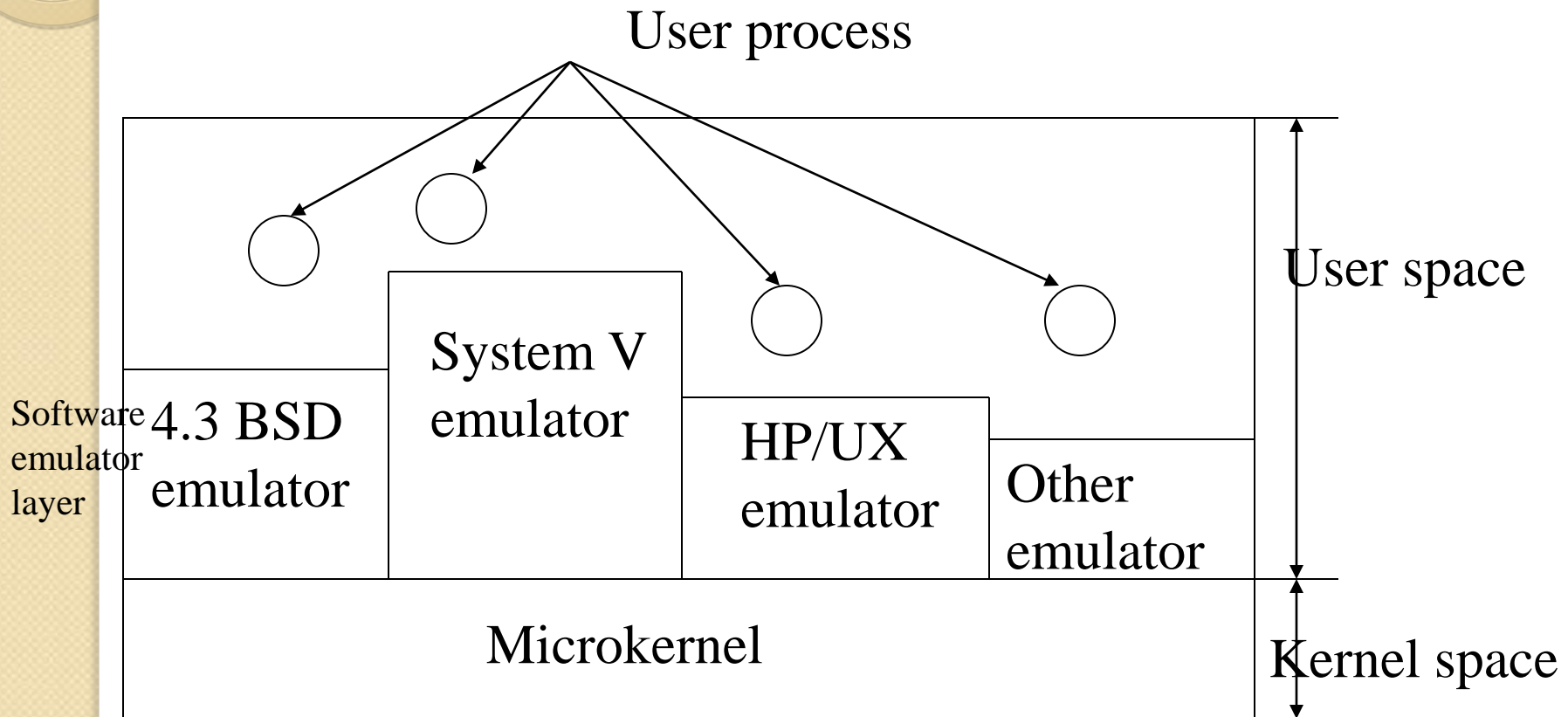
- By 1984 Accent was being used on 150 PERQs but it was clearly losing out to UNIX. This observation led Rashid to begin a third-generation operating systems project called Mach. Mach is compatible with UNIX, contains threads, multiprocessor support, and a virtual memory system.

- The first version of Mach was released in 1986 for the VAX 11/784, a four-CPU multiprocessor. Shortly thereafter, ports to the IBM PC/RT and Sun 3 were done. By 1987, Mach was also running on the Encore and Sequent multiprocessors. As of 1988, the Mach 2.5 kernel was large and monolithic, due to the presence of a large amount of Berkeley UNIX code in the kernel. In 1988, CMU removed all the Berkeley code from the kernel and put it in user space. What remained was a microkernel consisting of pure Mach. Mach is still under development.

Goals of Mach

1. Providing a base for building other operating systems (e.g., UNIX).
2. Supporting large sparse address spaces.
3. Allowing transparent access to network resources.
4. Exploiting parallelism in both the system and the applications.
5. Making Mach portable to a larger collection of machines.

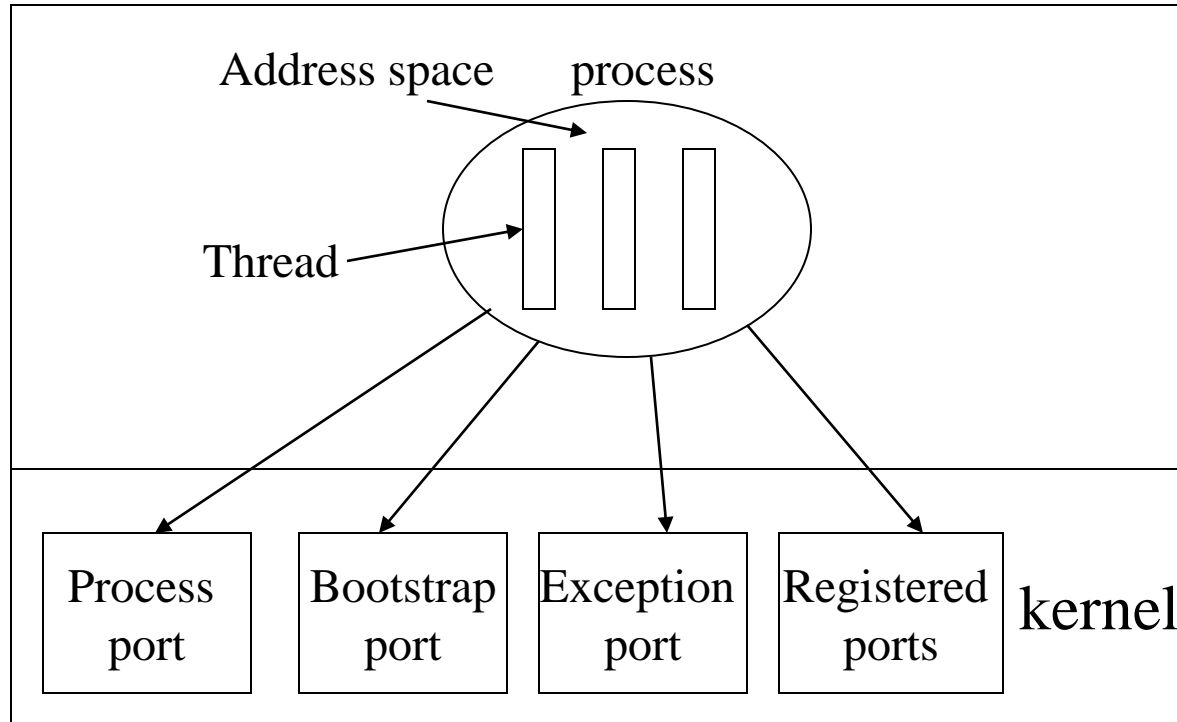
The Mach Microkernel



The kernel manages five principal abstractions:


1. Processes.
2. Threads.
3. Memory objects.
4. Ports.
5. Messages.

Process Management in Mach



Ports

- The process port is used to communicate with the kernel.
- The bootstrap port is used for initialization when a process starts up.
- The exception port is used to report exceptions caused by the process. Typical exceptions are division by zero and illegal instruction executed.
- The registered ports are normally used to provide a way for the process to communicate with standard system servers.

- 
- A process can be runnable or blocked.
 - If a process is runnable, those threads that are also runnable can be scheduled and run.
 - If a process is blocked, its threads may not run, no matter what state they are in.

Process Management Primitives

Create	Create a new process, inheriting certain properties
Terminate	Kill a specified process
Suspend	Increment suspend counter
Resume	Decrement suspend counter. If it is 0, unblock the process
Priority	Set the priority for current or future threads
Assign	Tell which processor new threads should run on
Info	Return information about execution time, memory usage, etc.
Threads	Return a list of the process' threads

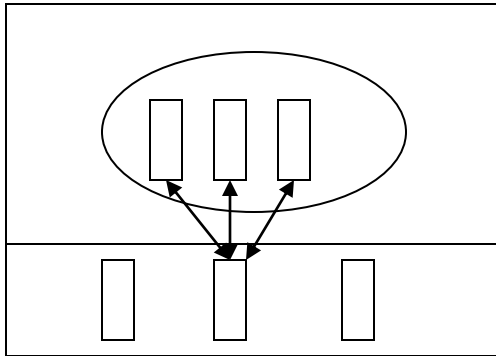
Threads

- Mach threads are managed by the kernel. Thread creation and destruction are done by the kernel.

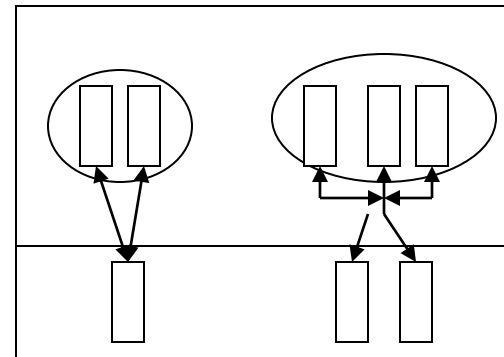
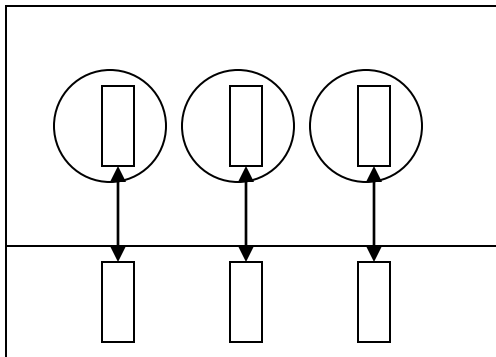
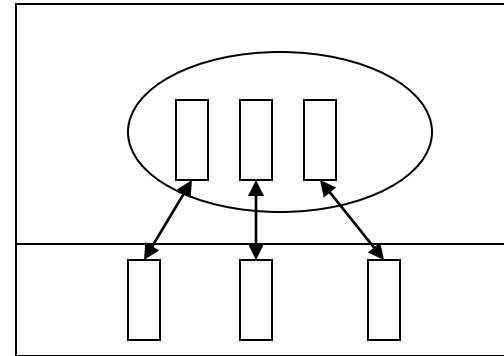
Fork	Create a new thread running the same code as the parent thread
Exit	Terminate the calling thread
Join	Suspend the caller until a specified thread exits
Detach	Announce that the thread will never be jointed (waited for)
Yield	Give up the CPU voluntarily
Self	Return the calling thread's identity to it

Implementation of C Threads in Mach

All C threads use one kernel thread.



Each C thread has its own kernel thread.



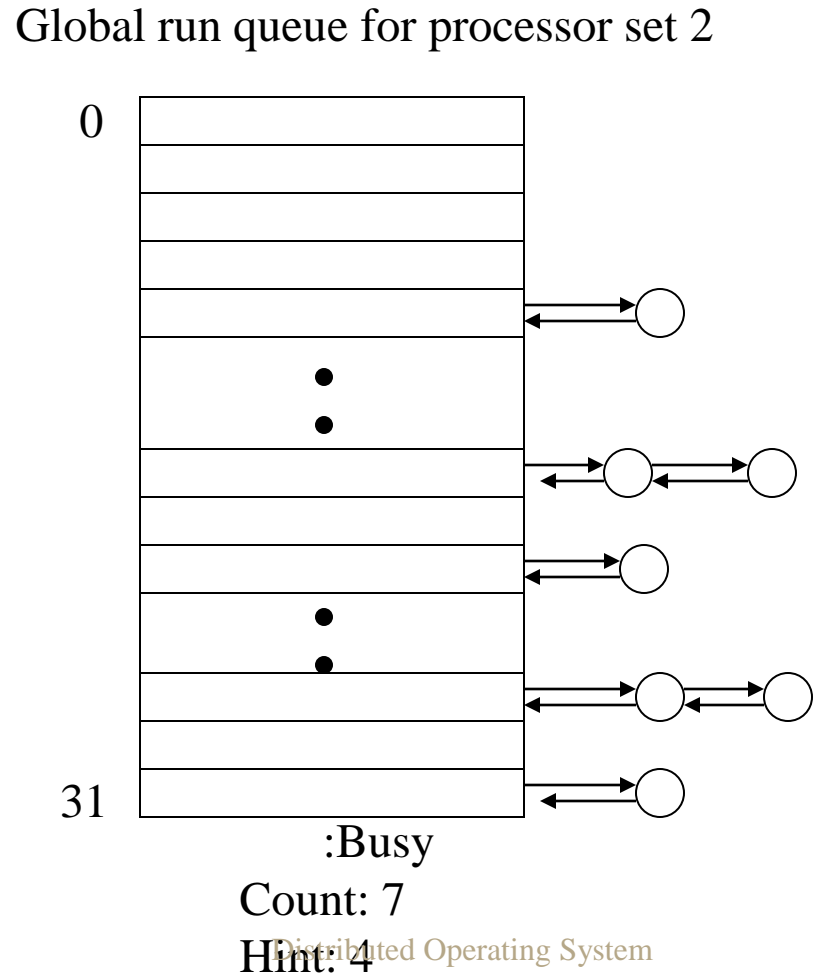
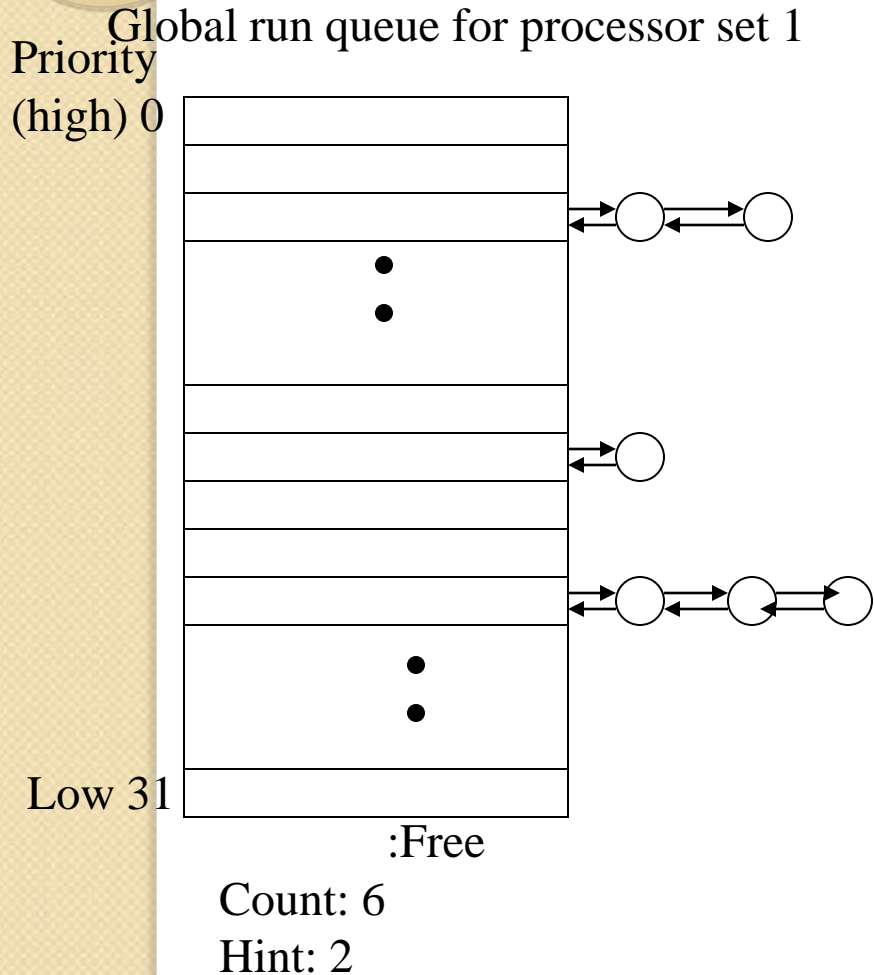
Each C thread has its own single-threaded process.

Arbitrary mapping of user threads to kernel threads.

Scheduling algorithm

- When a thread blocks, exits, or uses up its quantum, the CPU it is running on first looks on its local run queue to see if there are any active threads.
- If it is nonzero, run the highest-priority thread, starting at the queue specified by the hint.
- If the local run queue is empty, the same algorithm is applied to the global run queue. The global queue must be locked first.

Scheduling



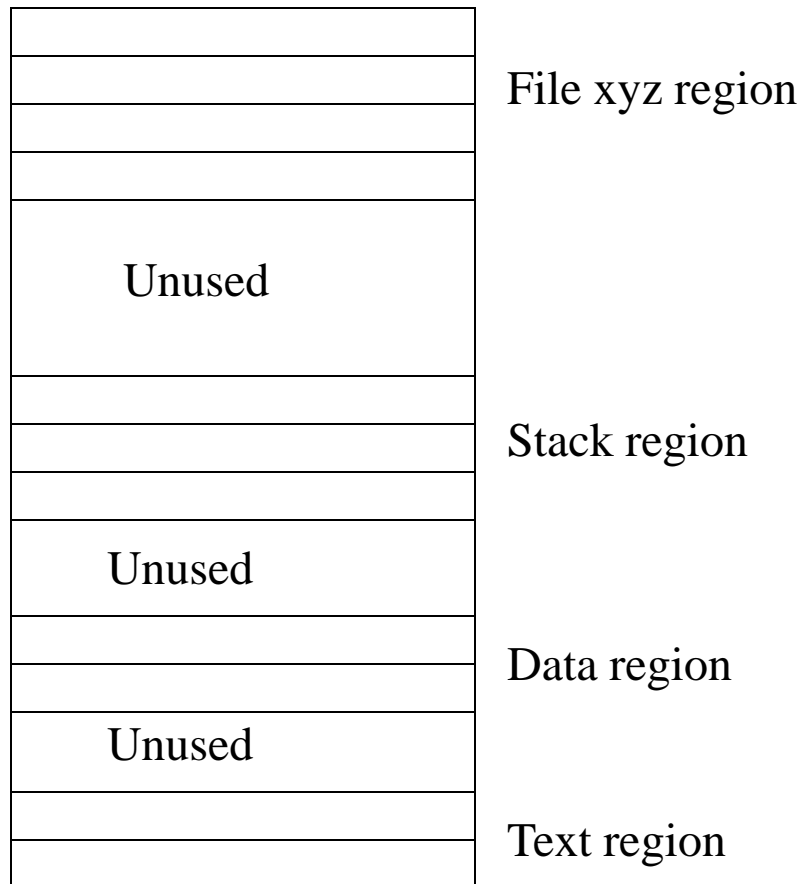
Memory Management in Mach

- Mach has a powerful, elaborate, and highly flexible memory management system based on paging.
- The code of Mach's memory management is split into three parts. The first part is the pmap module, which runs in the kernel and is concerned with managing the MMU.
- The second part, the machine-independent kernel code, is concerned with processing page faults, managing address maps, and replacing pages.
- The third part of the memory management code runs as a user process called a memory manager. It handles the logical part of the memory management system, primarily management of the backing store (disk).

Virtual Memory

- The conceptual model of memory that Mach user processes see is a large, linear virtual address space. The address space is supported by paging.
- A key concept relating to the use of virtual address space is the memory object. A memory object can be a page or a set of pages, but it can also be a file or other, more specialized data structure.

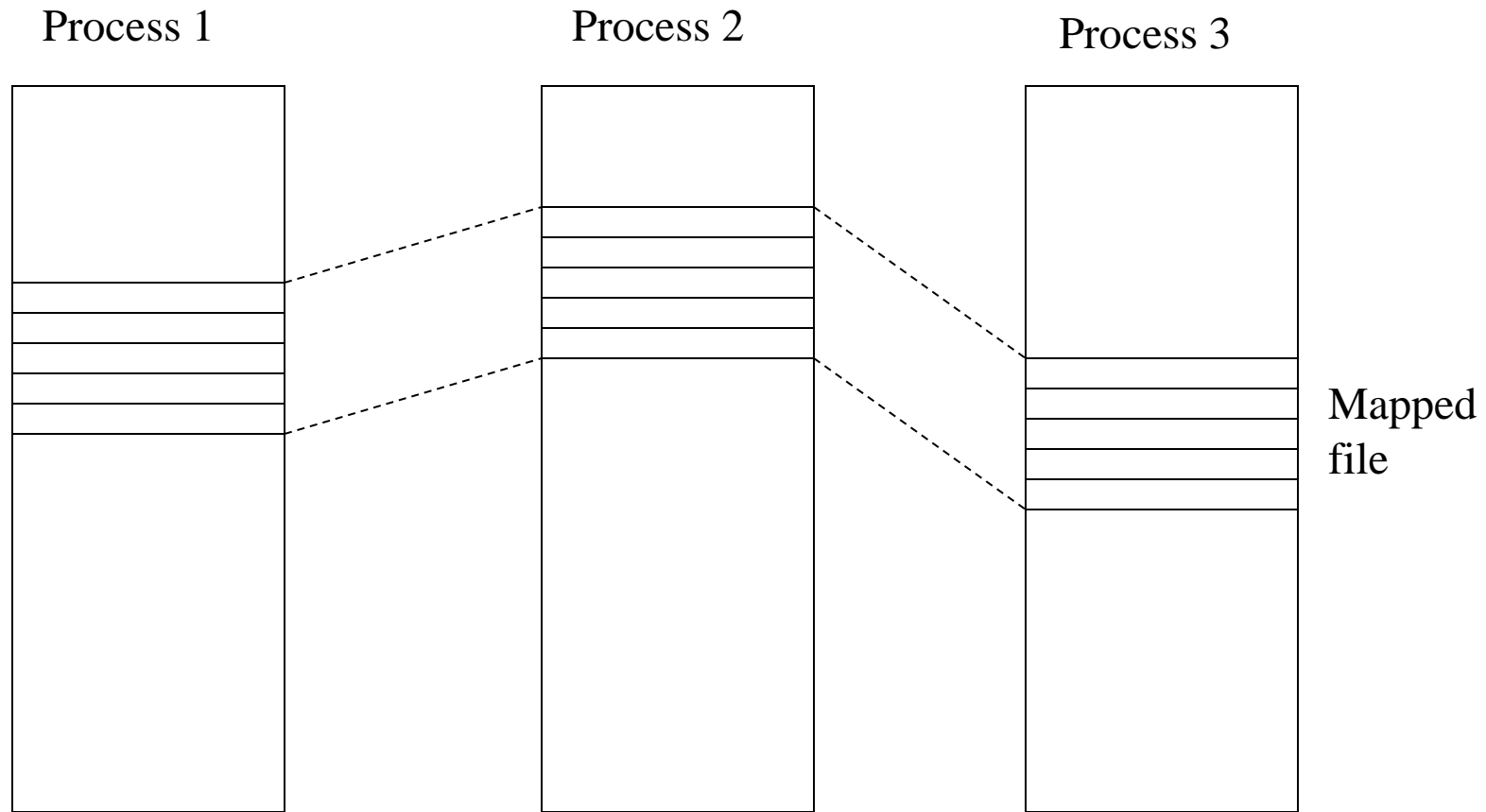
Address space with allocated regions, mapped objects, and unused addresses



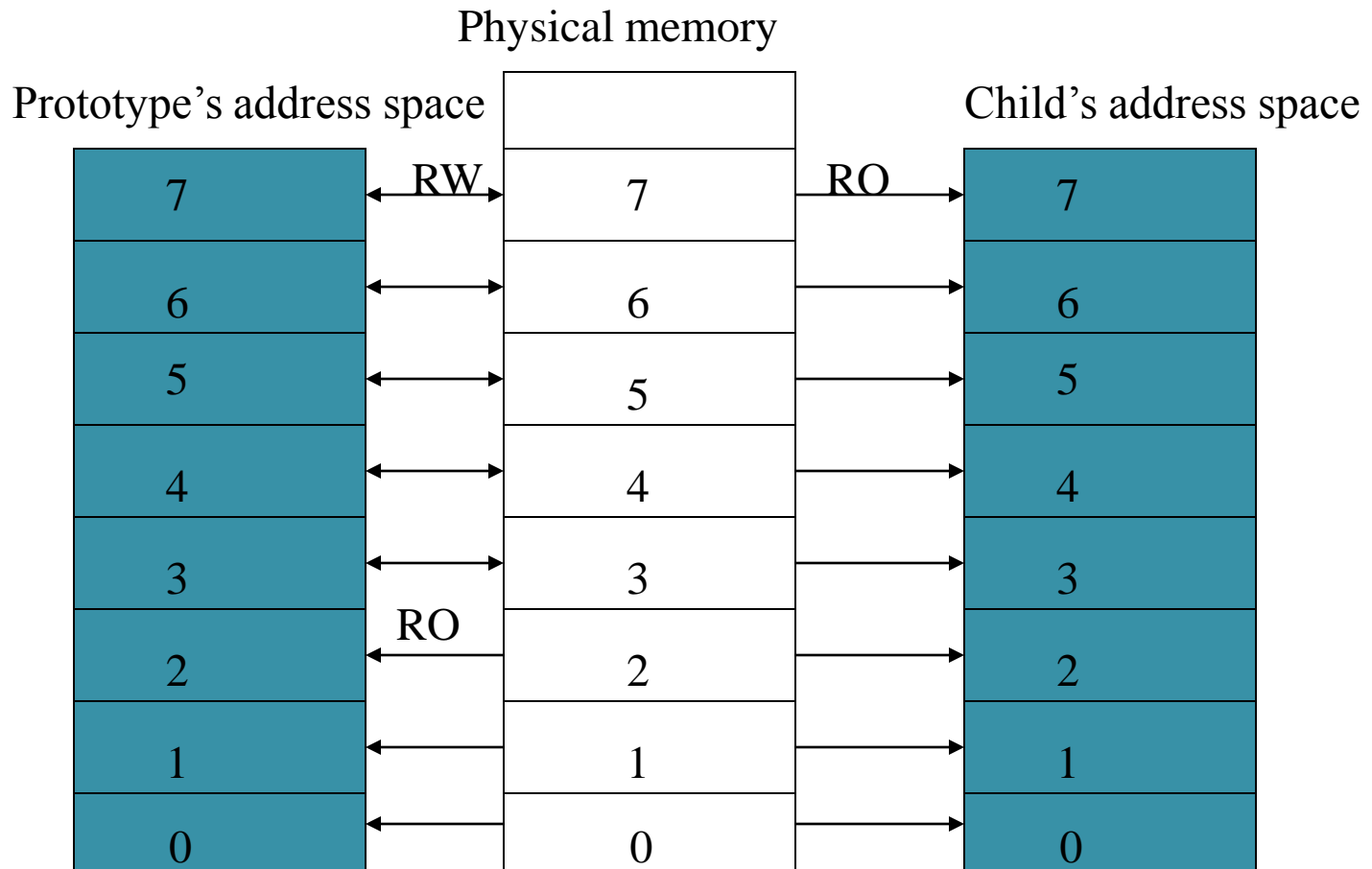
System calls for virtual address space manipulation

Allocate	Make a region of virtual address space usable
Deallocate	Invalidate a region of virtual address space
Map	Map a memory object into the virtual address space
Copy	Make a copy of a region at another virtual address
Inherit	Set the inheritance attribute for a region
Read	Read data from another process' virtual address space
Write	Write data to another process' virtual address space

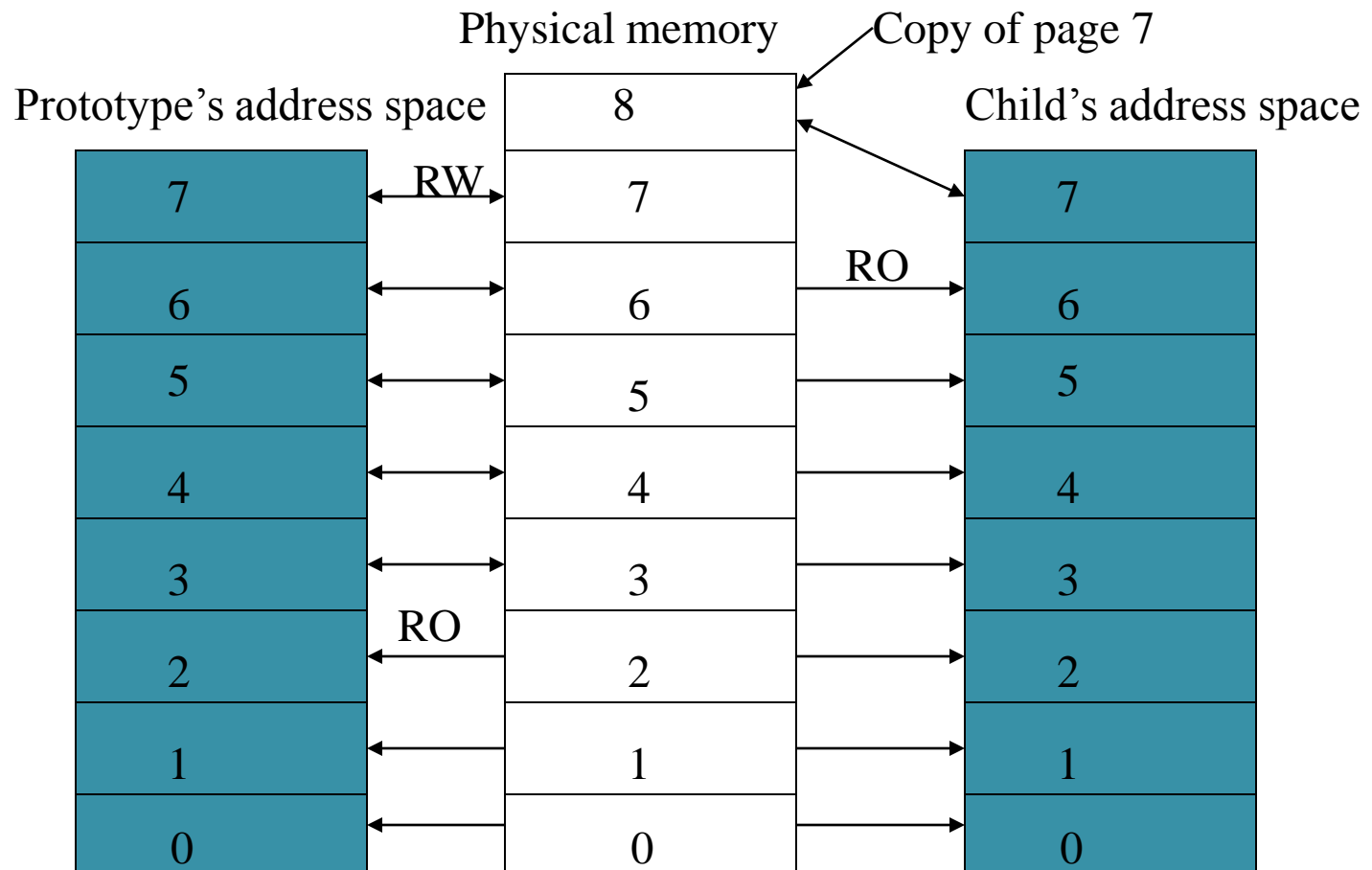
Memory Sharing



Operation of Copy-on-Write



Operation of Copy-on-Write



Advantages of Copy-on-write

1. some pages are read-only, so there is no need to copy them.
2. other pages may never be referenced, so they do not have to be copied.
3. still other pages may be writable, but the child may deallocate them rather than using them.

Disadvantages of Copy-on-write

1. the administration is more complicated.
2. requires multiple kernel traps, one for each page that is ultimately written.
3. does not work over a network.

External Memory Managers

- Each memory object that is mapped in a process' address space must have an external memory manager that controls it. Different classes of memory objects are handled by different memory managers.
- Three ports are needed to do the job.
- The object port, is created by the memory manager and will later be used by the kernel to inform the memory manager about page faults and other events relating to the object.
- The control port, is created by the kernel itself so that the memory manager can respond to these events.
- The name port, is used as a kind of name to identify the object.

Distributed Shared Memory in Mach

- The idea is to have a single, linear, virtual address space that is shared among processes running on computers that do not have any physical shared memory. When a thread references a page that it does not have, it causes a page fault. Eventually, the page is located and shipped to the faulting machine, where it is installed so that the thread can continue executing.

ASSIGNMENT

- Explain process management in distributed system in detail.